

Forward Kinematics of Stanford Robot Arm Using Denavit - Hartenberg Representation

MECE 617 Activity

Ivan John A. Naparota
University of San Carlos – Talamban Campus
Department of Electrical and Electronics Engineering
ivanjohnnaparota66@gmail.com

Abstract — this activity presents the Forward Kinematics of a Stanford Robot Arm using Denavit – Hartenberg Representation and with the aid of a numerical computing software, MATLAB.

Keywords – Forward Kinematics, Denavit-Hartenberg Representation, MATLAB, Stanford Robot

I. INTRODUCTION

Denavit–Hartenberg parameters (also called DH parameters) are the four parameters associated with a particular convention for attaching reference frames to the links of a spatial kinematic chain, or robot manipulator. Jacques Denavit and Richard Hartenberg introduced this convention in 1955 in order to standardize the coordinate frames for spatial linkages [2].

You just have to identify the parameters on each joint that is very needed in this convention, this will be defined further in the next parts of this paper. These parameters will then be used to perform the necessary operations that mainly involves multiplication of matrices for us to be able to obtain the position vector of the end effector, which is the main concern of forward kinematics.

The parameters of a Stanford Robot Arm is already presented on page 38 of the book “ROBOTICS: Control, Sensing, Vision, and Intelligence” by Fu, Gonzales, and Lee, these parameters will be the parameters to be used in this activity.

II. EQUATIONS AND MATLAB CODES

A. Formulas and Equations

For a specific i , we obtain the T matrix, $T = {}^0A_i$, which specifies the position and orientation of the endpoint of the manipulator with respect to the base coordinate system. Considering T matrix to be of the form [1]

$$T = \begin{bmatrix} \cos\theta & -\cos\alpha\sin\theta & \sin\alpha\sin\theta & a\cos\theta \\ \sin\theta & \cos\alpha\cos\theta & -\sin\alpha\cos\theta & a\sin\theta \\ 0 & \sin\alpha & \cos\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where:

d – depth along the previous joint’s z axis

θ – angle about the previous z to align its x with the new origin

r – distance along the rotated x axis

α – rotation about the new x axis to put z in its desired orientation

See figure 2,

Referring to page 38 of the book “Robotics: Control, Sensing, Vision, and Intelligence” by Fu, Gonzales, and Lee, established link coordinate systems for Stanford robot arm is hereby presented:

Stanford robot arm link coordinate parameters				
Joint i	θ_i	α_i	a_i	d_i
1	$\theta_1 = -90$	-90	0	d_1
2	$\theta_2 = -90$	90	0	d_2
3	-90	0	0	d_3
4	$\theta_4 = 0$	-90	0	0
5	$\theta_5 = 0$	90	0	0
6	$\theta_6 = 0$	0	0	d_6

And d’s depend on the inputs of the joints that will dictate the position of the end effector and the form of the Stanford robot.

And the homogeneous matrix 0T_i which specifies the location of the i th coordinate frame with respect to the base coordinate system is the chain product of successive coordinate transformation matrices of ${}^{i-1}A_i$, and is expressed as [1]

$${}^0T_i = {}^0A_1 {}^1A_2 \dots {}^{i-1}A_i = \begin{bmatrix} {}^0R_i & {}^0P_i \\ 0 & 1 \end{bmatrix}$$

Where:

P – Position vector of the hand

B. MATLAB Codes

```
clc;
clear all
close all

%depth
d1 = input('Input D1:');
d2 = input('Input D2:');
d3 = input('Input D3:');
d4 = 0;
d5 = 0;
d6 = input('Input D6');

%alpha
a1 = -90;
a2 = 90;
a3 = 0;
a4 = -90;
a5 = 90;
a6 = 0;

%length
l1 = 0;
l2 = 0;
l3 = 0;
l4 = 0;
l5 = 0;
l6 = 0;

%theta
t1 = -90;
t2 = -90;
t3 = -90;
t4 = 0;
t5 = 0;
t6 = 0;

%transformation matrices
transformation1 = [cosd(t1) -
(cosd(a1)*sind(t1)) sind(a1)*sind(t1)
l1*cosd(t1)
sind(t1) cosd(a1)*cosd(t1) -
(sind(a1)*cosd(t1)) l1*sind(t1)
0 sind(a1) cosd(a1) d1
0 0 0 1];

transformation2 = [cosd(t2) -
(cosd(a2)*sind(t2)) sind(a2)*sind(t2)
l2*cosd(t2)
```

```
sind(t2) cosd(a2)*cosd(t2) -
(sind(a2)*cosd(t2)) l2*sind(t2)
0 sind(a2) cosd(a2) d2
0 0 0 1];
```

```
transformation3 = [cosd(t3) -
(cosd(a3)*sind(t3)) sind(a3)*sind(t3)
l3*cosd(t3)
sind(t3) cosd(a3)*cosd(t3) -
(sind(a3)*cosd(t3)) l3*sind(t3)
0 sind(a3) cosd(a3) d3
0 0 0 1];
```

```
transformation4 = [cosd(t4) -
(cosd(a4)*sind(t4)) sind(a4)*sind(t4)
l4*cosd(t4)
sind(t4) cosd(a4)*cosd(t4) -
(sind(a4)*cosd(t4)) l4*sind(t4)
0 sind(a4) cosd(a4) d4
0 0 0 1];
```

```
transformation5 = [cosd(t5) -
(cosd(a5)*sind(t5)) sind(a5)*sind(t5)
l5*cosd(t5)
sind(t5) cosd(a5)*cosd(t5) -
(sind(a5)*cosd(t5)) l5*sind(t5)
0 sind(a5) cosd(a5) d5
0 0 0 1];
```

```
transformation6 = [cosd(t6) -
(cosd(a6)*sind(t6)) sind(a6)*sind(t6)
l6*cosd(t6)
sind(t6) cosd(a6)*cosd(t6) -
(sind(a6)*cosd(t6)) l6*sind(t6)
0 sind(a2) cosd(a2) d6
0 0 0 1];
```

```
%multiply
multiply1 = transformation1 *
transformation2;
multiply2 = transformation1 *
transformation2 * transformation3;
multiply3 = transformation1 *
transformation2 * transformation3 *
transformation4;
multiply4 = transformation1 *
transformation2 * transformation3 *
transformation4 * transformation5;
multiply = transformation1 *
transformation2 * transformation3 *
transformation4 * transformation5 *
transformation6;
```

```
%output
output = [multiply(1,4) multiply(2,4)
multiply(3,4)]';
```

```
disp('End Effector is Located on
Point(x,y,z):');
disp(num2str(output, '%.2f'))
```

```
%plot
line([0 0],[0 0],[-660.4
0],'LineWidth',2)
line([0 multiply1(1,4)],[0
multiply1(2,4)],[0 0],'LineWidth',2,
'color','r');
line([multiply1(1,4)
multiply2(1,4)],[multiply1(2,4)
multiply2(2,4)],[0
multiply2(3,4)'],'LineWidth',2,
'color','g')
line([multiply2(1,4)
multiply3(1,4)],[multiply2(2,4)
multiply3(2,4)],[multiply2(3,4)
multiply3(3,4)'],'LineWidth',2,
'color','y')
line([multiply3(1,4)
multiply4(1,4)],[multiply3(2,4)
multiply4(2,4)],[multiply3(3,4)
multiply4(3,4)'],'LineWidth',2,
'color','b')
line([multiply4(1,4)
multiply(1,4)],[multiply4(2,4)
multiply(2,4)],[multiply4(3,4)
multiply(3,4)'],'LineWidth',2,
'color','k')
axis([-1000 1000 -1000 1000 -1000 1000])
grid on
```

```
%message
disp('Type [StanfordForwardDH] in
Command Window to Try Again');
```

IV. RESULTS AND DISCUSSIONS

As shown in Figure 1, the program will ask for the input d's that will define the position of the end effector and form of our Stanford robot, d's are the only inputs we have because the other parameters are already defined as presented above.

For these given d's,

Input D1: 600

Input D2: 600

Input D3: 600

Input D6: 600

And considering the defined parameters presented above,

End effector is located in (x,y,z) point 600, 1200, 600), and the plot is shown in Figure 3.

It is observable that the plot in Figure 3 shows the same form with the ones presented on page 38 of the book "ROBOTICS: Control, Sensing, Vision, and Intelligence" that was shown on Figure 2 of this document, this tells us that our Forward Kinematics solution is correct.

V. CONCLUSION

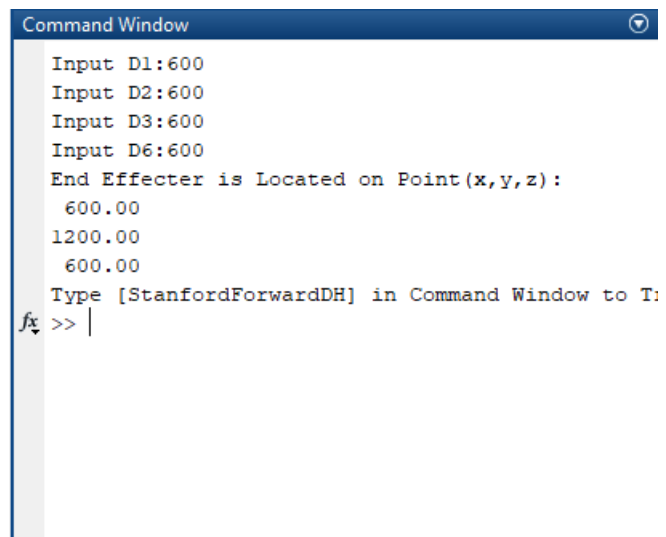
Denavit – Hartenberg representation is capable of solving the forward kinematics of a Stanford robot arm, you just have to define the necessary parameters for this convention.

Also, MATLAB's computing prowess would help a lot to do the necessary operations without the need for manual solutions.

VI. REFERENCES

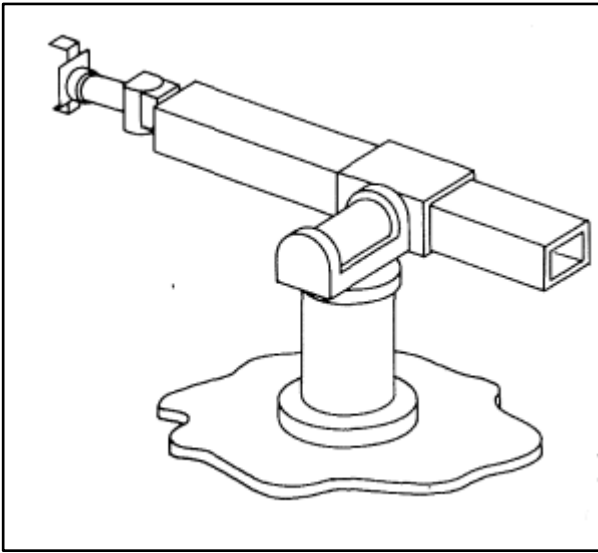
1. Fu, K. S., Gonzales, R. C., Lee, C. S. G., ROBOTICS : Control, Sensing, Vision, and Intelligence.
2. https://en.wikipedia.org/wiki/Denavit%E2%80%933Hartenberg_parameters

VII. FIGURES



```
Command Window
Input D1:600
Input D2:600
Input D3:600
Input D6:600
End Effector is Located on Point(x,y,z):
 600.00
1200.00
 600.00
Type [StanfordForwardDH] in Command Window to T:
fx >> |
```

Figure 1 MATLAB Command Window



Fu, K. S., Gonzales, R. C., .Lee, C. S. G., ROBOTICS : Control, Sensing, Vision, and Intelligence.

Figure 2 Stanford Robot

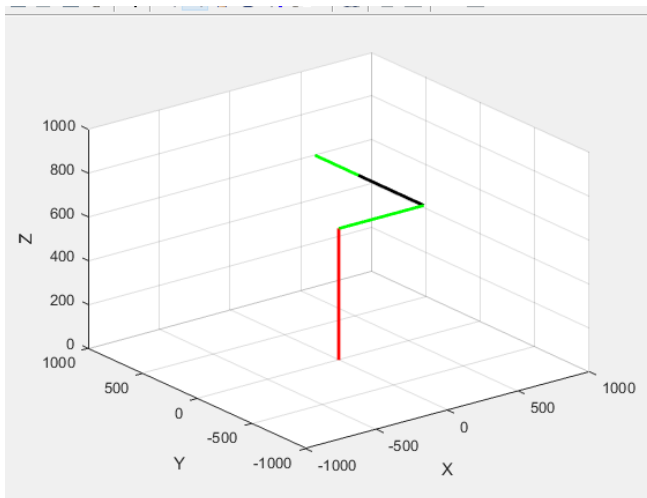


Figure 3. Plot of Stanford Robot Arm